Tensor product of kernel models

Omar De la Cruz C., Alex Barnett, Hua Tang and Susan Holmes

December 10, 2010

Introduction

Kernels allow us to discover and exploit non-linear structure in data simply by applying linear multivariate methods, based on inner products, substituting the kernel for those inner products ("kernel trick," based on Moore–Aronszajn's Theorem).

Introduction

Kernels allow us to discover and exploit non-linear structure in data simply by applying linear multivariate methods, based on inner products, substituting the kernel for those inner products ("kernel trick," based on Moore–Aronszajn's Theorem).

This approach has been very successful, especially in classification (e.g., SVMs).

Introduction

Kernels allow us to discover and exploit non-linear structure in data simply by applying linear multivariate methods, based on inner products, substituting the kernel for those inner products ("kernel trick," based on Moore–Aronszajn's Theorem).

This approach has been very successful, especially in classification (e.g., SVMs).

However, some challenges arise in exploratory methods, like Kernel PCA: interpretation of the results is often tricky.

Horseshoes

Consider the classical Iris data set [Fisher 1936; Anderson 1935] (only the 50 observations from the *Setosa* species).

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

$\vdots$

Horseshoes

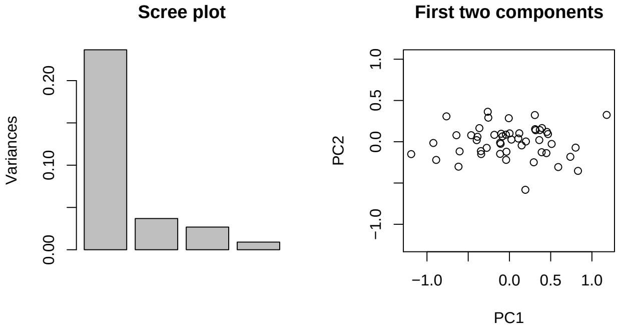
Consider the classical Iris data set [Fisher 1936; Anderson 1935] (only the 50 observations from the *Setosa* species).

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

⋮

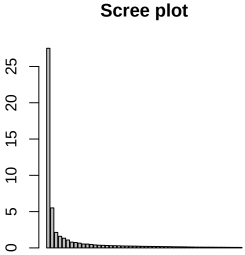
Straightforward PCA shows a dominant component (likely corresponding to overall size):

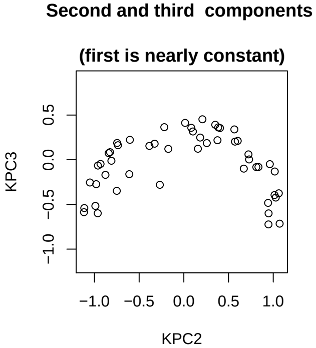
PCA for the Iris Setosa data



Kernel version

Consider now the kernel $K(x,y) = e^{-||x-y||}$





Voting records in US Congress (House)

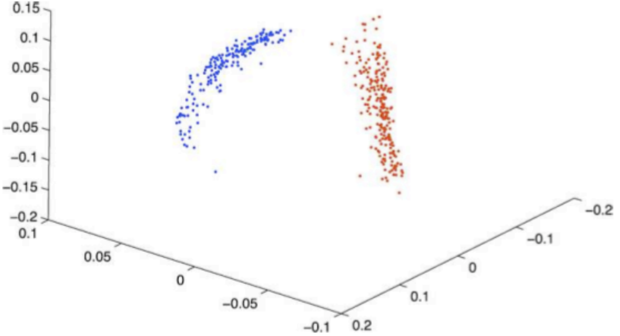


FIG. 1. *3-Dimensional MDS output of legislators based on the 2005 U.S. House roll call votes. Color has been added to indicate the party affiliation of each Representative.*

Figure 1 from [Diaconis–Goel–Holmes, 2008].

By a *kernel* we (loosely) mean an $n \times n$ similarity matrix K. It might be computed from:

1. Data table X, by a suitable comparison of rows with each other.
2. Distance matrix D, by some transformation (e.g., entry-wise)
3. D might be an intermediate step between X and K.

Examples

1. Linear kernel (PCA): $K = XX'$, usually after centering X by columns.

Examples

1. Linear kernel (PCA): $K = XX'$, usually after centering X by columns.
2. Multidimensional scaling (MDS): Start from distance-like D, doubly-center, and multiply by $-1/2$.

Examples

1. Linear kernel (PCA): $K = XX'$, usually after centering X by columns.
2. Multidimensional scaling (MDS): Start from distance-like D, doubly-center, and multiply by $-1/2$.
3. Exponential kernel: $K = \exp(-D/\lambda)$, computed entry-wise.

Examples

1. Linear kernel (PCA): $K = XX'$, usually after centering X by columns.
2. Multidimensional scaling (MDS): Start from distance-like D, doubly-center, and multiply by $-1/2$.
3. Exponential kernel: $K = \exp(-D/\lambda)$, computed entry-wise.
4. RBF kernel: $K = \exp(-D^2/\sigma^2)$, computed entry-wise.

Examples

1. Linear kernel (PCA): $K = XX'$, usually after centering X by columns.
2. Multidimensional scaling (MDS): Start from distance-like D, doubly-center, and multiply by $-1/2$.
3. Exponential kernel: $K = \exp(-D/\lambda)$, computed entry-wise.
4. RBF kernel: $K = \exp(-D^2/\sigma^2)$, computed entry-wise.
5. Adjacency kernel: Build a graph with n nodes; add an edge between any two individuals closer than a given threshold; K is the adjacency matrix.

Examples

1. Linear kernel (PCA): $K = XX'$, usually after centering X by columns.
2. Multidimensional scaling (MDS): Start from distance-like D, doubly-center, and multiply by $-1/2$.
3. Exponential kernel: $K = \exp(-D/\lambda)$, computed entry-wise.
4. RBF kernel: $K = \exp(-D^2/\sigma^2)$, computed entry-wise.
5. Adjacency kernel: Build a graph with n nodes; add an edge between any two individuals closer than a given threshold; K is the adjacency matrix.
6. Graph Laplacian kernel: from the adjacency kernel subtract the diagonal matrix of degrees.

Examples

1. Linear kernel (PCA): $K = XX'$, usually after centering X by columns.
2. Multidimensional scaling (MDS): Start from distance-like D, doubly-center, and multiply by $-1/2$.
3. Exponential kernel: $K = \exp(-D/\lambda)$, computed entry-wise.
4. RBF kernel: $K = \exp(-D^2/\sigma^2)$, computed entry-wise.
5. Adjacency kernel: Build a graph with n nodes; add an edge between any two individuals closer than a given threshold; K is the adjacency matrix.
6. Graph Laplacian kernel: from the adjacency kernel subtract the diagonal matrix of degrees.

Examples

1. Linear kernel (PCA): $K = XX'$, usually after centering X by columns.
2. Multidimensional scaling (MDS): Start from distance-like D, doubly-center, and multiply by $-1/2$.
3. Exponential kernel: $K = \exp(-D/\lambda)$, computed entry-wise.
4. RBF kernel: $K = \exp(-D^2/\sigma^2)$, computed entry-wise.
5. Adjacency kernel: Build a graph with n nodes; add an edge between any two individuals closer than a given threshold; K is the adjacency matrix.
6. Graph Laplacian kernel: from the adjacency kernel subtract the diagonal matrix of degrees.

Sometimes the kernels might be scaled and/or centered by rows and columns.

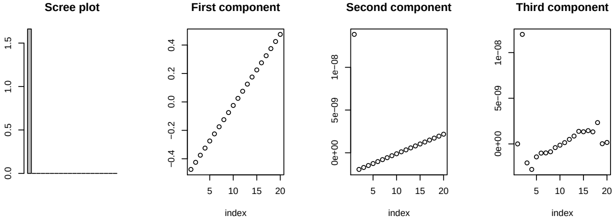
Understanding horseshoes

The approach taken in [Diaconis–Goel–Holmes, 2008] was to analyze in detail a simple case: The data points lie on a one-dimensional grid.

Understanding horseshoes

The approach taken in [Diaconis–Goel–Holmes, 2008] was to analyze in detail a simple case: The data points lie on a one-dimensional grid.

When analyzed using PCA, the result is (of course) trivial:

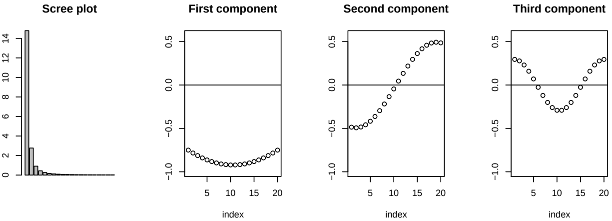


One-dimensional grid, exponential kernel

Taking now the kernel $K(x, y) = e^{-|x-y|}$ we obtain:

One-dimensional grid, exponential kernel

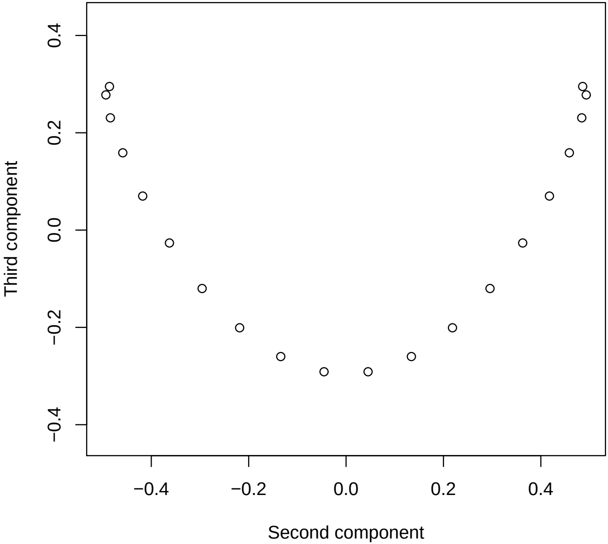
Taking now the kernel $K(x, y) = e^{-|x-y|}$ we obtain:



[DGH] showed that these eigenvectors are approximately described by trigonometric functions, and when $n \to \infty$, these eigenvectors converge to the eigenfunctions of the integral operator on $\mathcal{L}^2([0, 1])$ defined by the kernel.

Here is the horseshoe!

When we plot the second vs. the third components against each other we get:



The scree plot might suggest that the second (or higher) components are relevant, *but these might only be elements of a basis for the space of continuous functions on* $[0, 1]$.

The scree plot might suggest that the second (or higher) components are relevant, *but these might only be elements of a basis for the space of continuous functions on* $[0, 1]$.

The kernel changes the intrinsic geometry of the space (in this case $[0, 1]$), *but not its topology*.

The scree plot might suggest that the second (or higher) components are relevant, *but these might only be elements of a basis for the space of continuous functions on* $[0, 1]$.

The kernel changes the intrinsic geometry of the space (in this case $[0, 1]$), *but not its topology*.

The extra basis elements are called in to "help relieve the stress" of embedding a non-euclidean space into euclidean space.

The scree plot might suggest that the second (or higher) components are relevant, *but these might only be elements of a basis for the space of continuous functions on* $[0, 1]$.

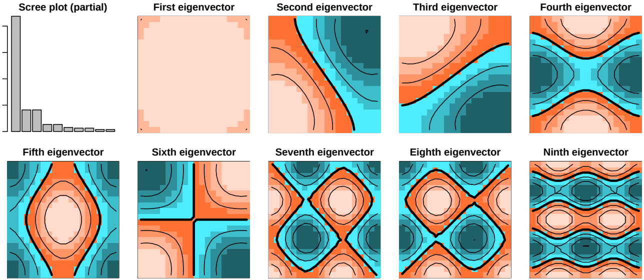
The kernel changes the intrinsic geometry of the space (in this case $[0, 1]$), *but not its topology*.

The extra basis elements are called in to "help relieve the stress" of embedding a non-euclidean space into euclidean space.

We do not conclude that the data has intrinsic dimension 2, or higher.

Two dimensions: Grid case

In the same spirit, we explore now the eigenvectors of kernels obtained from a two-dimensional grid of points.



Scree plot and first nine eigenfunctions for the exponential kernel on the 2d grid.

Proposition

The exponential kernel is separable, if we use the $\mathfrak{L}^1$ ("city block") distance; that is,

$$K_2\left((x,y),(w,z)\right) = K_1(x,w)K_1(y,z).$$

Proposition

The exponential kernel is separable, if we use the $\mathfrak{L}^1$ ("city block") distance; that is,

$$K_2\left((x, y), (w, z)\right) = K_1(x, w)K_1(y, z).$$

Proposition

$$\mathbf{K}_2 = \mathbf{K}_1 \otimes \mathbf{K}_1,$$

where $\otimes$ is the Kronecker product of matrices

Proposition

The exponential kernel is separable, if we use the $\mathfrak{L}^1$ ("city block") distance; that is,

$$K_2\left((x,y),(w,z)\right) = K_1(x,w)K_1(y,z).$$

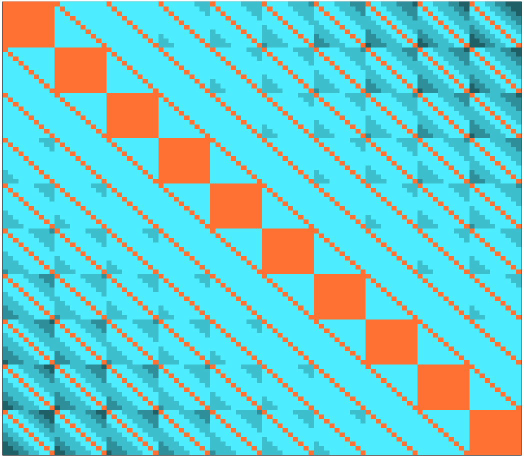
Proposition

$$\mathbf{K}_2 = \mathbf{K}_1 \otimes \mathbf{K}_1,$$

where $\otimes$ is the Kronecker product of matrices

It should be noted that, since $\mathbf{K}_1$ is Toeplitz, $\mathbf{K}_2$ is *block Toeplitz with Toeplitz blocks* (it is not Toeplitz, though).

Kronecker product of 1d kernel by itself



Proposition

If v, u are eigenvectors for A, B, with eigenvalues λ, γ, respectively, then $v \otimes u$ is an eigenvector for $A \otimes B$ with eigenvalue $\lambda\gamma$.

Proposition

If v, u are eigenvectors for A, B, with eigenvalues λ, γ, respectively, then $v \otimes u$ is an eigenvector for $A \otimes B$ with eigenvalue $\lambda\gamma$.

Proposition

Fixing a full set of eigenvectors for $\mathbf{K}_1$, the eigenvalues and eigenvectors of $\mathbf{K}_2$ described in the previous proposition form a full eigendecomposition of $\mathbf{K}_2$. Thus, $\mathbf{K}_2$ has N eigenvalues with multiplicity 1 (the squares of the eigenvalues of $\mathbf{K}_1$) and $\binom{N}{2}$ eigenvalues with multiplicity 2.

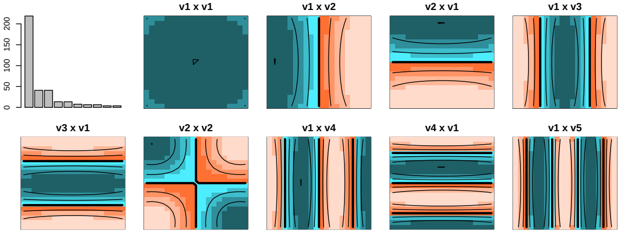
Proposition

If v, u are eigenvectors for A, B, with eigenvalues λ, γ, respectively, then $v \otimes u$ is an eigenvector for $A \otimes B$ with eigenvalue $\lambda\gamma$.

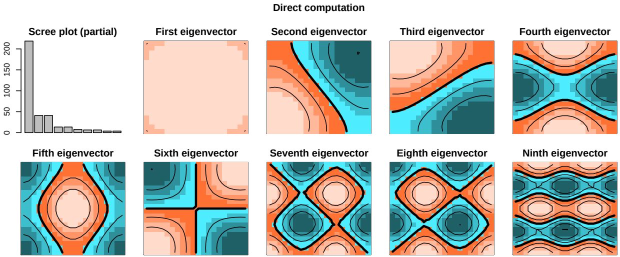
Proposition

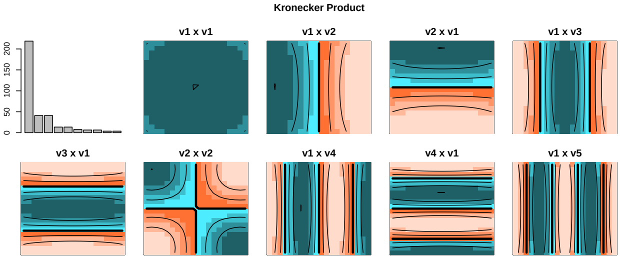
Fixing a full set of eigenvectors for $\mathbf{K}_1$, the eigenvalues and eigenvectors of $\mathbf{K}_2$ described in the previous proposition form a full eigendecomposition of $\mathbf{K}_2$. Thus, $\mathbf{K}_2$ has N eigenvalues with multiplicity 1 (the squares of the eigenvalues of $\mathbf{K}_1$) and $\binom{N}{2}$ eigenvalues with multiplicity 2.

In spite of these results, the eigenfunctions in the figure above *do not seem to result from multiplying the one-dimensional eigenfunctions*. Indeed, the products look as follows:

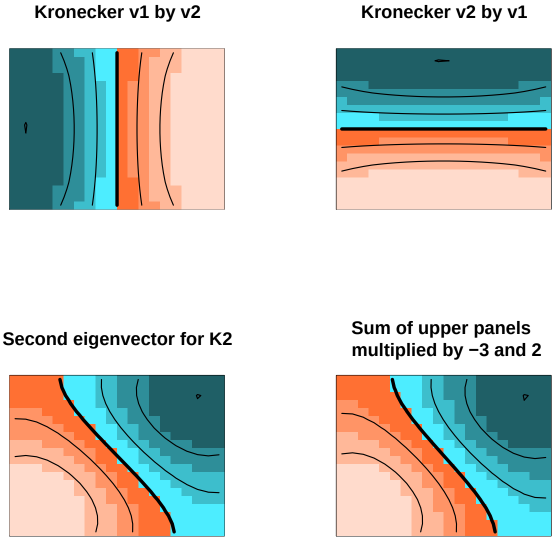


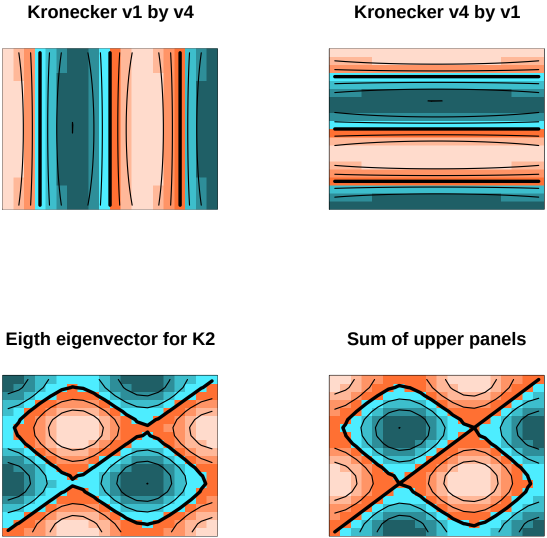
Eigenvalues for the 2d exponential kernel, and eigenfunctions obtained as outer products of eigenfunctions of the 1d exponential kernel.


Direct computation
Scree plot (partial)
0
50
100
150
200
First eigenvector
Second eigenvector
Third eigenvector
Fourth eigenvector
Fifth eigenvector
Sixth eigenvector
Seventh eigenvector
Eighth eigenvector
Ninth eigenvector


Kronecker Product
0
50
100
150
200
v1 x v1
v1 x v2
v2 x v1
v1 x v3
v3 x v1
v2 x v2
v1 x v4
v4 x v1
v1 x v5

The mystery is solved by noticing that for the repeated eigenvalues the 2-element eigenbasis chosen by the eigendecomposition software can be arbitrarily rotated.


Kronecker v1 by v2
Kronecker v2 by v1
Second eigenvector for K2
Sum of upper panels
multiplied by −3 and 2


Kronecker v1 by v4
Kronecker v4 by v1
Eigth eigenvector for K2
Sum of upper panels

2D grid as a graph

If we think of the 1D grid as a graph G_1, with edges between immediately adjacent points, we can obtain a 2D grid in at least two ways:

2D grid as a graph

If we think of the 1D grid as a graph G_1, with edges between immediately adjacent points, we can obtain a 2D grid in at least two ways:

Cartesian product of graphs:
Let $\langle V_1, E_1\rangle, \langle V_2, E_2\rangle$ be graphs; the cartesian product graph is:

$$\langle V_1, E_1\rangle \square \langle V_2, E_2\rangle = \langle V_1 \times V_2, E\rangle$$

with E defined by

$$((u, v), (w, z)) \in E \text{ iff}$$

$$(u = w \text{ and } (v, z) \in E_2) \text{ or } (v = z \text{ and } (u, w) \in E_1) \,.$$

2D grid as a graph

If we think of the 1D grid as a graph G_1, with edges between immediately adjacent points, we can obtain a 2D grid in at least two ways:

Cartesian product of graphs:
Let $\langle V_1, E_1\rangle, \langle V_2, E_2\rangle$ be graphs; the cartesian product graph is:

$$\langle V_1, E_1\rangle \Box \langle V_2, E_2\rangle = \langle V_1 \times V_2, E\rangle$$

with E defined by

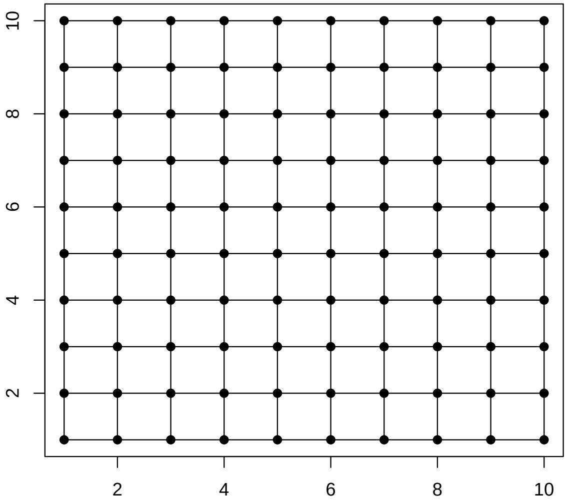
$$((u, v), (w, z)) \in E \text{ iff}$$

$$(u = w \text{ and } (v, z) \in E_2) \text{ or } (v = z \text{ and } (u, w) \in E_1).$$

From this definition we can find a formula for the adjacency matrices:

$$A = A_1 \otimes I_{|V_2|} + I_{|V_1|} \otimes A_2$$

(the expression on the right is known as the “Kronecker sum” of A_1 and A_2, denoted $A_1 \oplus A_2$).



2
4
6
8
10
2
4
6
8
10

The graph distance (length of the shortest path joining two vertices) on G_{2C} is the $\mathcal{L}^1$ distance.

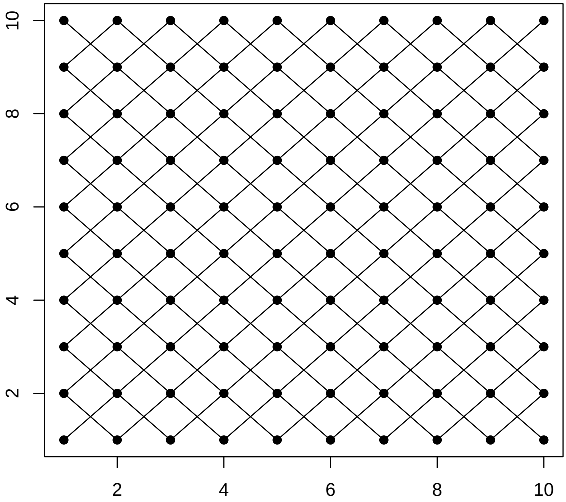
Tensor product of graphs
Let $\langle V_1, E_1\rangle, \langle V_2, E_2\rangle$ be graphs; the tensor product graph is:

$$\langle V_1, E_1\rangle \otimes \langle V_2, E_2\rangle = \langle V_1 \times V_2, E\rangle$$

with E defined by

$$((u,v),(w,z)) \in E \text{ iff } (u,w) \in E_1 \text{ and } (v,z) \in E_2.$$

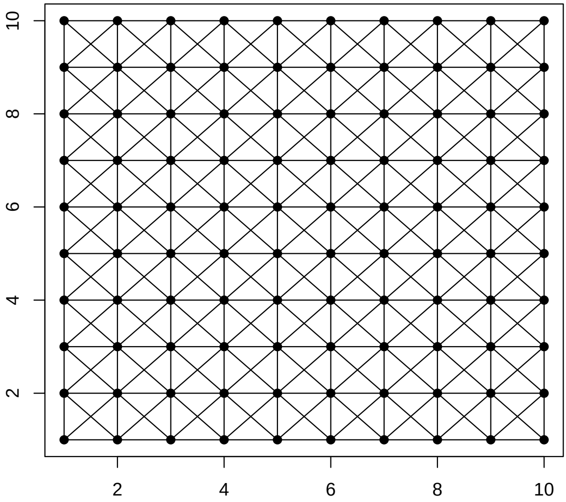
It follows from this definition that $A = A_1 \otimes A_2$ (Kronecker product).
Consider the tensor product $G_1 \otimes G_1$. This has only diagonal edges between vertices that are $\sqrt{2}$ units apart:


10
8
6
4
2
2
4
6
8
10

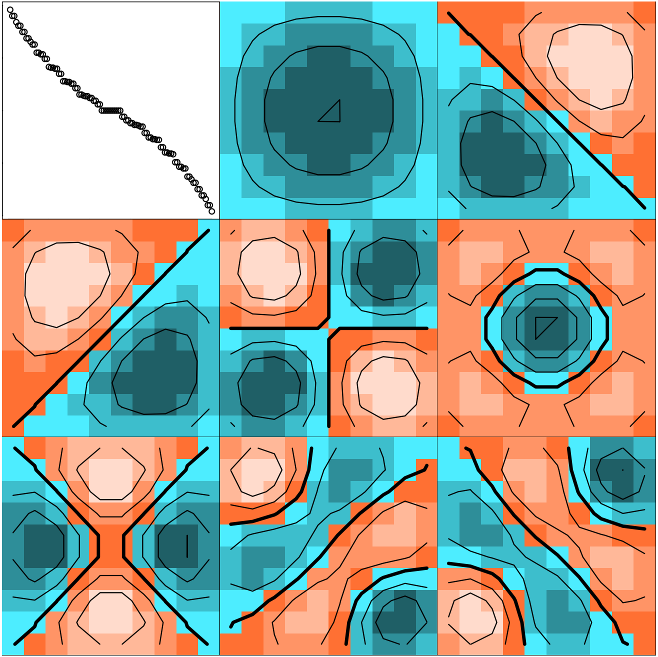
This is clearly not a reasonable answer; this graph is in fact disconnected (as it happens with every tensor product of two bipartite graphs). However, this can be remedied if we add loops at each vertex of G_1; call that graph G_1^+.

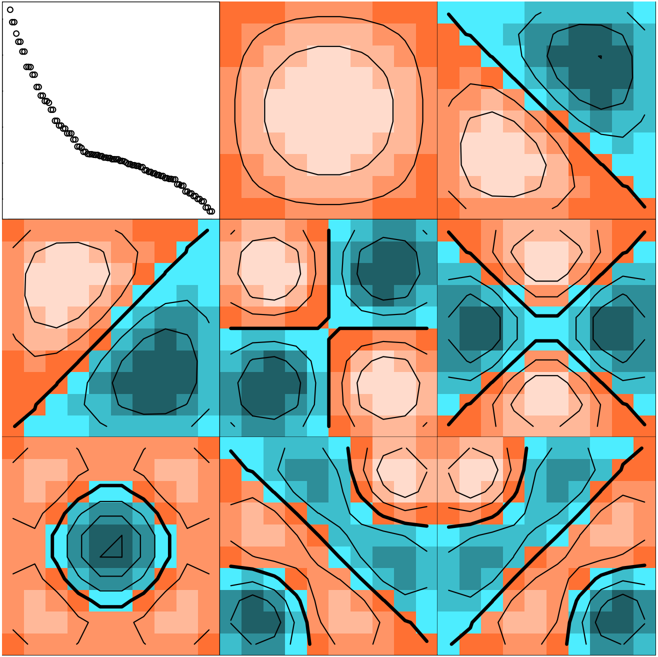
This is clearly not a reasonable answer; this graph is in fact disconnected (as it happens with every tensor product of two bipartite graphs). However, this can be remedied if we add loops at each vertex of G_1; call that graph G_1^+.

Define $G_{2T} = G_1^+ \otimes G_1^+$. Then G_{2T} has vertical and horizontal edges between vertices that are one unit away, plus diagonal edges between points $\sqrt{2}$ units apart.


10
8
6
4
2
2
4
6
8
10

Now look at the eigenfunctions for the adjacency kernel in each case:





The eigenfunctions look very similar! (up to changes of sign). This is indeed true, due to the following fact:

The eigenfunctions look very similar! (up to changes of sign). This is indeed true, due to the following fact:

Proposition

If v, u are eigenvectors for A, B, with eigenvalues λ, γ, respectively, then $v \otimes u$ is an eigenvector for $A \oplus B$ with eigenvalue $\lambda + \gamma$.

The eigenvectors are the same as for the tensor product.

The eigenvectors are the same as for the tensor product.

The eigenvalues are different because now we add them instead of multiplying them.

The eigenvectors are the same as for the tensor product.

The eigenvalues are different because now we add them instead of multiplying them.

Still, this leads to eigendecompositions with the same N eigenspaces with dimension 1 and $\binom{N}{2}$ eigenspaces with dimension 2, and *these eigenspaces are the same for both kernels*. The eigenvalues are different, and they need not appear in the same order (except for the largest eigenvalue).

The eigenvectors are the same as for the tensor product.

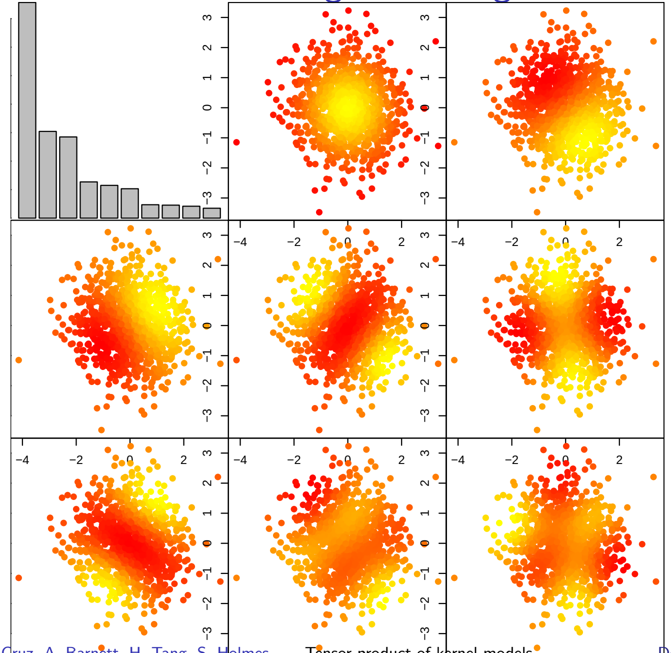
The eigenvalues are different because now we add them instead of multiplying them.

Still, this leads to eigendecompositions with the same N eigenspaces with dimension 1 and $\binom{N}{2}$ eigenspaces with dimension 2, and *these eigenspaces are the same for both kernels*. The eigenvalues are different, and they need not appear in the same order (except for the largest eigenvalue).

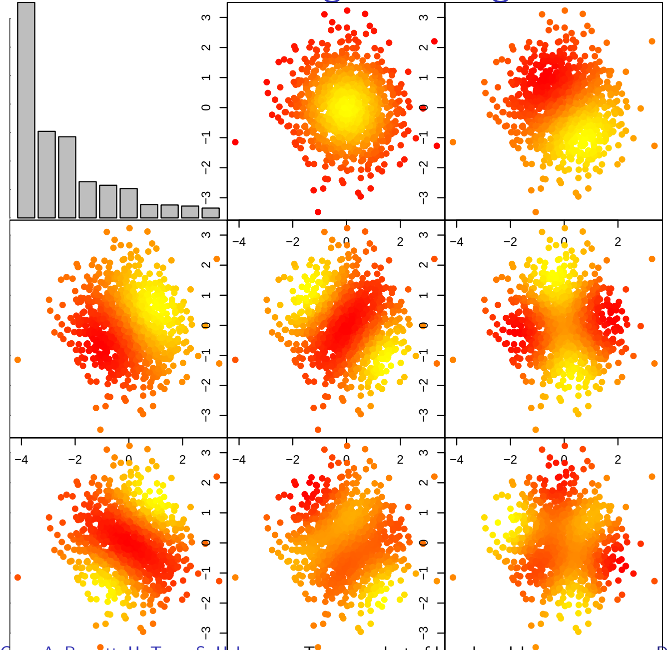
Furthermore, each of those eigenspaces admits a basis formed with Kronecker products of eigenvectors for A_1 (or A_1^+, since they are the same).

The obvious pattern is the presence of nodal domains (regions where the entries of the eigenvectors are all positive or all negative). These domains become smaller, and the alternating patterns more complex, as the eigenvalues become smaller.

What about non-rectangular, non-grid sets of points?



What about non-rectangular, non-grid sets of points?



What about non-rectangular, non-grid sets of points?

To deal with both limitations, it is useful to look at kernels in a statistical model setting.

Kernel Models

The statistical model includes:

1. The landscape space $\mathcal{X}$, where the samples come from, together with a notion of smoothness for real-valued functions on $\mathcal{X}$, which is given by the choice of a kernel function $\mathcal{K} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, symmetric and of positive type.
2. A sampling probability measure P on $\mathcal{X}$; the n observations in the study are assumed to have been sampled i.i.d. according to P.

In fact, what one is choosing is a RKHS of functions on $\mathcal{X}$ whose elements will be those functions declared to be smooth on $\mathcal{X}$.

Operators Associated with the Kernel

Assuming there is a reference probability distribution Q on $\mathcal{X}$ (for example, uniform), we have two smoothing operators acting on $\mathcal{L}^2(\mathcal{X}, P)$:

$$T_{\mathcal{K}} : f \mapsto \int_{\mathcal{X}} f(y)\mathcal{K}(\cdot, y)\, dQ(y) \quad \text{and} \quad S_{\mathcal{K}} : f \mapsto \int_{\mathcal{X}} f(y)\mathcal{K}(\cdot, y)\, dP(y).$$

Remark: Since we want to learn about $\mathcal{X}$, we are usually more interested in $T_{\mathcal{K}}$; but, unless we also estimate P, and adjust accordingly, we will be estimating $S_{\mathcal{K}}$ instead.

Operators Associated with the Kernel

Assuming there is a reference probability distribution Q on $\mathcal{X}$ (for example, uniform), we have two smoothing operators acting on $\mathcal{L}^2(\mathcal{X}, P)$:

$$T_{\mathcal{K}} : f \mapsto \int_{\mathcal{X}} f(y)\mathcal{K}(\cdot, y)\, dQ(y) \quad \text{and} \quad S_{\mathcal{K}} : f \mapsto \int_{\mathcal{X}} f(y)\mathcal{K}(\cdot, y)\, dP(y).$$

Remark: Since we want to learn about $\mathcal{X}$, we are usually more interested in $T_{\mathcal{K}}$; but, unless we also estimate P, and adjust accordingly, we will be estimating $S_{\mathcal{K}}$ instead.

[Rosasco–Belkin–Vito 2010] and [Smale–Zhou 2009] consider this framework, and prove large sample results for the eigenfunctions of these operators and the eigenvectors from the discrete versions obtained by sampling.

Binning

As a simple way to approximate the models and the corresponding operators, we can use binning.

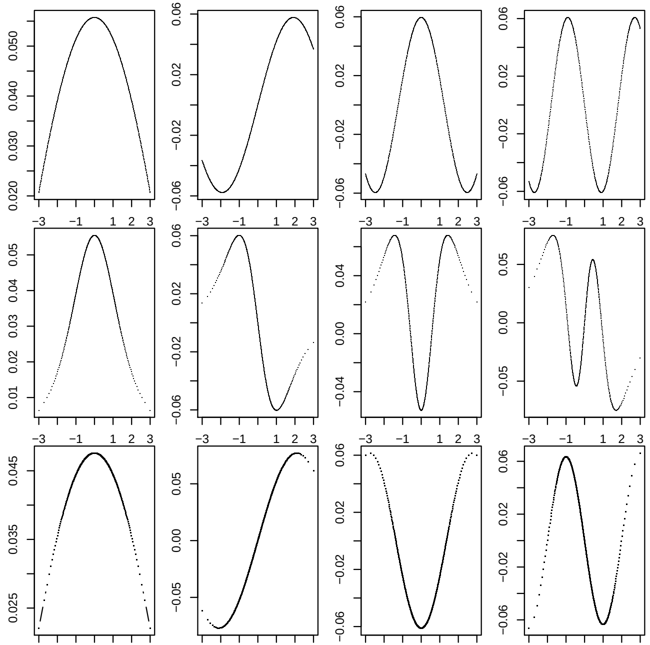
Consider the one-dimensional case: $\mathcal{X}$ is an interval.

If we split $\mathcal{X}$ into equal bins, the relative frequencies of samples in each bin provide an estimate of the density dP (histogram).

Then we can consider the matrix $\mathbf{K}$ obtained from the bin centers. Matrix multiplication is a discrete approximation to the operators $T_{\mathcal{K}}$ and $S_{\mathcal{K}}$.

The density estimate comes in as a set of **weights**.

Eigenvectors for exponential kernel on $[-3, 3]$



Computed with a regular grid of points

Computed with points sampled from the standard normal distribution

Computed with points sampled from the standard normal distribution, with **weights** given by the standard normal density.

Tensor products of Kernel Models

We can now consider the product of two models $(\mathcal{X}, \mathcal{K}_{\mathcal{X}}, P_{\mathcal{X}}), (\mathcal{Y}, \mathcal{K}_{\mathcal{Y}}, P_{\mathcal{Y}})$ to obtain a model with higher intrinsic dimension:

The landscape is taken to be the cartesian product $\mathcal{X} \times \mathcal{Y}$

The kernel is the Kronecker product $\mathcal{K}_{\mathcal{X}} \otimes \mathcal{K}_{\mathcal{Y}}$

The sampling probability distribution is obtained as the product measure $P_{\mathcal{X}} \times P_{\mathcal{Y}}$ (thus implying the assumption of independence of the sampling probabilities).

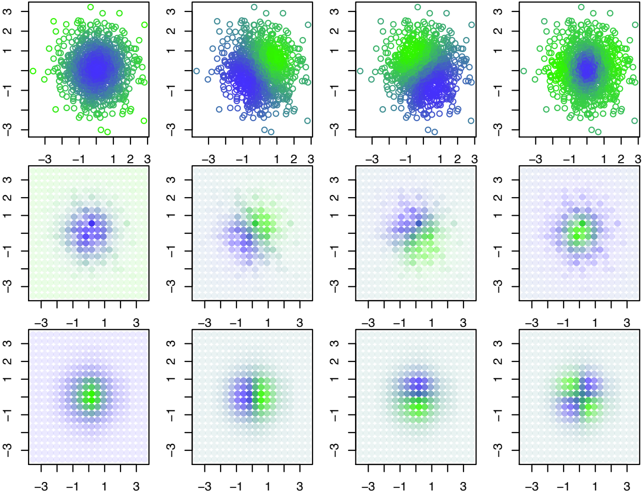
The landscape is still “rectangular,” since it is a cartesian product, but the observations come from a probability distribution that is not necessarily uniform on this rectangle (for example, it could be a bivariate normal distribution, if the sampling probability distributions on the factor models were univariate normal).

Two-way Binning

To approximate the models and the corresponding operators, we can use two-way binning

But considering the model as a tensor product of lower dimensional models, we can perform estimation on the margins, to obtain more stable results.

Eigenvector computed on points from a bivariate normal



Computed directly with the distances between points.

Computed using square bins, with the corresponding density estimate.

Computed using bins on the margins, and then combining them by tensor product.

Questions

- Higher dimensions
- Automated factoring
- Application: Finding inherent dimension in genotype data in mixed yet heterogeneous populations.

Conclusions

The model described, and the correspondence between kernels and operators, opens the door to many techniques from, e.g., Functional Analysis.

Here we described how it can be used to implement the Kronecker product of kernel matrices in a reasonable way for exploratory analysis, when the data do not lie on a grid.